Chapter 1 - Understanding MSMQ

This chapter introduces the terms and concepts you must understand to install and administer MSMQ. It begins with a terminology overview, covers the conceptual topics of interest to administrators, and concludes with a section covering some common message-queuing business scenarios.

MSMQ Terminology Overview

MSMQ applications communicate using *messages*. A message is simply a unit of information or data being sent between computers. The message can contain text or binary data. *Transactional* messages can be used to pair the sending or receiving of any message with an action in another operation. Using transactional messages ensures that the unit of work is carried out as an atomic operation—that is, the operation succeeds or fails as a whole. Transactional messages can also be used to ensure that a message is delivered only once and to ensure that all messages sent from one computer to another are delivered in order. Positive and negative acknowledgments can be used to confirm that messages reached or were retrieved from the destination queue.

MSMQ supports two delivery methods: *express* and *recoverable*. Choosing between express and recoverable delivery is a matter of trading performance and resource use for reliability and failure recovery. In general, express messages use fewer resources and are faster than recoverable messages. However, express messages cannot be recovered if the computer storing the memory-mapped message files fails. Recoverable messages use more resources and are slower than express messages, but can be recovered regardless of which computer fails.

MSMQ uses public and private *queues* to store and forward messages. All MSMQ queues, regardless of their function, can be manipulated with the same MSMQ functions. This includes the special *journal*, *dead letter*, *transactional dead letter*, *administration*, *system*, and *report* queues. Each of the queues is simply a standard MSMQ queue used for a specific purpose. For more information on the MSMQ API, see the Microsoft Message Queue Server Software Development Kit (MSMQ SDK).

Queue quotas and *computer quotas* specify the cumulative limit for messages in a queue or in all queues on a computer. The queue and computer quotas are based on size and can be set independently. When a queue quota is reached, messages can no longer be sent to the queue until one or more messages are removed from the queue. When a computer quota is reached, messages can no longer be sent to any queues on the computer until one or more messages are removed from one of the queues.

MSMQ routes messages to public queues based on the sum of each message's *message* and the message's destingation *queue priority*. MSMQ routes messages to private queues based on only the message's *message priority*.

MSMQ supports *dependent clients*, *independent clients*, and *servers*. Both independent clients and servers run the MSMQ service and can communicate asynchronously. MSMQ dependent clients require synchronous access to an MSMQ server.

Some MSMQ servers hold copies of the *MSMQ information store (MQIS)* database. The MQIS is a distributed database that holds enterprise topology, enterprise settings, computer information, and queue information. MSMQ-based applications can query the MQIS to find queues and get queue properties.

All computers operate within one MSMQ *enterprise*. The enterprise is divided into *sites*, where communication between any two computers is fast and inexpensive. Sites are connected through *site links*. *Site-link costs* define the cost of sending messages between sites. Computers running in MSMQ communicate over *connected networks (CNs)*. A CN is a collection of computers in which any two computers can communicate directly. MSMQ servers designated as *in-routing servers (InRSs)*, *out-routing servers (OutRSs)*, and *site gates* can be used to control the flow of messages and provide *session concentration*. MSMQ servers take all these factors into account when *routing* messages within your MSMQ enterprise.

Top of page

Topology and Connectivity

Before installing or configuring MSMQ, you must understand the following terms:

- MSMQ Enterprise

- MSMQ Sites
- MSMQ Connected Networks

MSMQ Enterprise

In MSMQ, all computers that run MSMQ belong to one enterprise and access information from the same distributed database, called the MSMQ information store(MQIS). To simplify administration and for compatibility with future versions of Windows NT Server, you should not install multiple enterprises within your organization. Issues relating to security and isolating the use of MSMQ between groups within your organization can be addressed using MSMQ security features.

However, if you choose to have more than one enterprise within a company, or want to exchange MSMQ messages with another company (for example, over the Internet) you can still do so. For information on sending messages between enterprises, see the MSMQ SDK.

MSMQ Sites

A site is a physical collection of computers in which communication between any two computers is fast and inexpensive. Site boundaries usually parallel the physical location of the computers (for example, all computers within a building). However, not all computers in a site have to run the same protocol, and computers in the same site may not be able to directly communicate with each other.

Sites are connected to other sites through communication links called *site links*. Inter-site routing is the process of sending messages between sites on these links. MSMQ calculates inter-site routing based on relative numbers that administrators assign to site links. These numbers, called *site-link costs*, represent the cost of communication of that link.

Although establishing site boundaries in MSMQ is fairly simple, additional factors should be considered for compatibility with the site object in future versions of Windows NT Server. For more information, see "Topology" in Chapter 6, "Deploying MSMQ."

Site-Link Costs

MSMQ calculates inter-site routing based on the *cost* of each site link. Site-link costs are defined using relative numbers between zero and 999,999. It is up to the administrator to define the relative cost of routing between sites. An administrator typically balances cost with delay (the speed of one link versus another).

You set site-link costs when you install new sites. If you have only two sites, choose any value above zero. If you have three or more sites, and the cost of routing between sites is more or less equal, use the same value for each site link. However, if you have three or more sites and if the cost of routing between sites is not equal, use site-link costs to define the difference in the routing costs. For example, suppose you have two sites called A and B in one city connected by a high-speed link, and one site called C, which is overseas and connected to site B by a low-speed link. Define the site-link cost between A and B as 1, and between B and C as 2.

A site-link cost of zero indicates that the two sites are not connected.

For more information on defining site links, see "MSMQ Routing" later in this chapter; "Installing a PSC" in Chapter 2, "Installing MSMQ"; and MSMQ Explorer Help.

MSMQ Connected Networks

Within MSMQ, a connected network (CN) is a collection of computers in which any two computers can communicate directly. The computers within a CN must support the same protocol and must be able to establish a session. A computer can belong to multiple CNs, and CNs can span sites. However, all computers in a physical local area network (all computers monitoring the same broadcasts) that use the same protocol (IP or IPX) must belong to the same CN.

When you define a CN, you are simply defining a label. When you install an MSMQ server, you associate each network address on the computer with the appropriate CNs. These CNs form logical groupings of computers that can communicate directly. MSMQ dependent clients use their supporting server CNs, and MSMQ independent clients automatically determine their CNs.

Tip When you define CNs for your enterprise, use meaningful labels so that administrators can easily choose a CN from a list when overriding the default CN settings. For more information, see "Naming Conventions" in Chapter 6, "Deploying MSMQ."

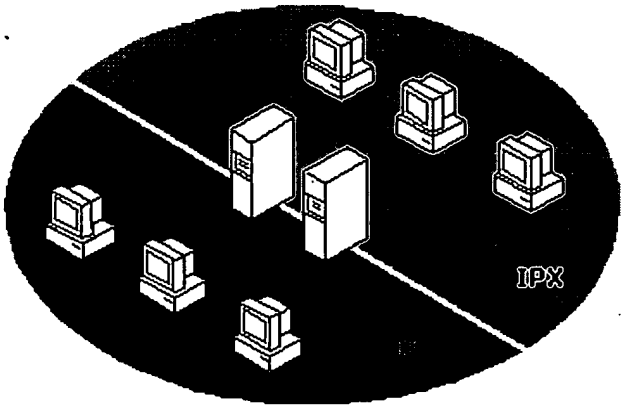

Figure 1.1

In most cases, the CN layout in an enterprise mirrors the protocol use. For example, in Figure 1.1, half the client computers run IP, the other half run IPX, and the servers run both IP and IPX. The MSMQ clients running IP belong to an IP CN and the MSMQ clients running IPX belong to the IPX CN. The servers belong to both CNs. However, CNs are not always defined to mirror protocol use. For example, you need two CNs when you have two separate IPX networks (subnets) on the same local area network.

Figure 1.2 shows an enterprise with two sites. The enterprise has two CNs, with one of the CNs spanning both sites. The servers belong to both the IP CN and the IPX CN to support the routing of MSMQ messages between CNs.

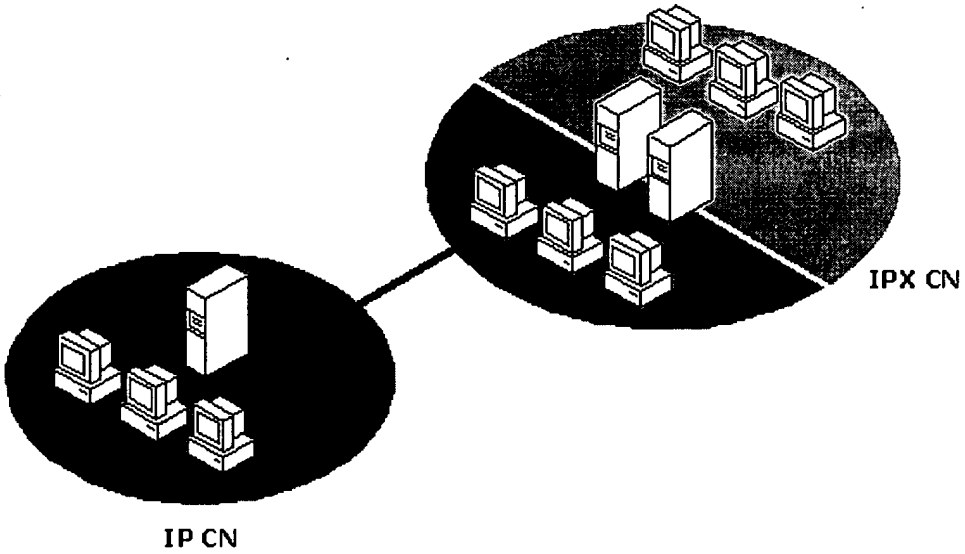

Figure 1.2

In this example, at least one of the two servers in the rightmost site must belong to both CNs in order for computers in the IPX CN to communicate with computers in the IP CN.

↑ Top of page

MSMQ Servers

MSMQ uses four server types to control message queuing:

- Primary enterprise controller (PEC)
- Primary site controller (PSC)
- Backup site controller (BSC)
- MSMQ routing server

Each of these servers must be installed on a computer running Windows NT Server, Enterprise Edition. The specific

functions of these servers are explained in the following sections. Your MSMQ enterprise must contain one PEC. The installation of other server types is optional, depending on your network topology, the size of your MSMQ enterprise, and the types of MSMQ-based applications you develop. You can also configure Windows NT Remote Access Service (RAS) servers to support remote MSMQ independent clients. You can do this by installing one of the four MSMQ server types on the RAS server, or by installing only the MSMQ RAS Connectivity Service on the RAS server.

MSMQ Explorer uses a variety of icons to display the different types of computers. To see a table that defines the computer, queue, and message icons, click the Help button on the MSMQ toolbar and click anywhere in the MSMQ enterprise.

For more information on determining the appropriate number of MSMQ servers for your MSMQ network and their location and configuration, see Chapter 6, "Deploying MSMQ." For more information on installing the various MSMQ servers, see Chapter 2, "Installing MSMQ."

MSMQ Server Licensing Considerations

MSMQ does not limit the number of client/server sessions. However, MSMQ does count the number of Windows NT client access licenses (CALs) and restricts concurrent MSMQ client sessions accordingly. MSMQ supports both Per Server and Per Seat licensing.

For more information on CALs, see *Microsoft Windows NT* Server *Version 4.0 Concepts and Planning*, Chapter 12, "Licensing and License Manager."

Primary Enterprise Controller

Administrators install one PEC on an MSMQ network. The PEC functions as a PSC for one site. The PEC holds information about the enterprise configuration and the certification keys (used in authenticating messages) in a database, and also functions as an MSMQ routing server. You must install a PEC before you can install any PSCs.

Primary Site Controller

You install one PSC for each additional site in your MSMQ network. The PEC functions as the site controller for the initial site you create. The PSC holds information about the computers and queues in the site in a database, and also functions as an MSMQ routing server. Although you do not have to install a PSC in each site (as defined in a preceding section, "MSMQ Sites"), it is highly recommended that you do.

Using MSMQ in a Site without a PSC

When you use MSMQ in a site without a PSC (or any other MSMQ server), you lose many of the benefits of MSMQ, including the efficient and persistent routing of messages. When the site is disconnected from the PEC site, independent clients within the site cannot locate queues or create queues, and each client uses additional resources as it attempts to resend undeliverable messages. MSMQ dependent clients can be used within the site only when the site controller is online.

Backup Site Controller

A site does not require a BSC. However, one or more BSCs should be installed at each site to provide load balancing and failure recovery, should the PSC or PEC fail. The BSC holds a read-only replica of the PSC or PEC database and also functions as an MSMQ routing server. You must install a PEC or PSC before you can install any BSCs.

MSMQ Routing Server

MSMQ routing servers support dynamic routing and intermediate store-and-forward message queuing. They allow computers using different protocols to communicate and can be used to provide session concentration. Unlike BSCs, MSMQ routing servers do not hold a read-only replica of the PSC or PEC database. You must install a PEC before you can install any MSMQ routing servers.

Because every PEC, PSC, and BSC functions as an MSMQ routing server, the number of additional MSMQ routing servers you install depends on your MSMQ connectivity requirements (number of dependent clients, independent clients, sites, CNs, session concentration needs, and message volume). As a minimum, you should strategically install enough MSMQ routing servers to allow messages to reach target queues through different servers.

MSMQ Connector Server

Any MSMQ server can also function as an MSMQ connector server. MSMQ connector servers allow MSMQ-based

applications to communicate with computers (called *foreign computers*) that either use other messaging systems or support the MSMQ functions on hardware not supported by Windows 95, Windows NT Workstation, or Windows NT Server. MSMQ connector servers use *foreign CNs* and *foreign queues* to communicate with foreign computers, as shown in Figure 1.3. The Level 8 Systems MSMQ message queuing product is an example of an MSMQ connector server.

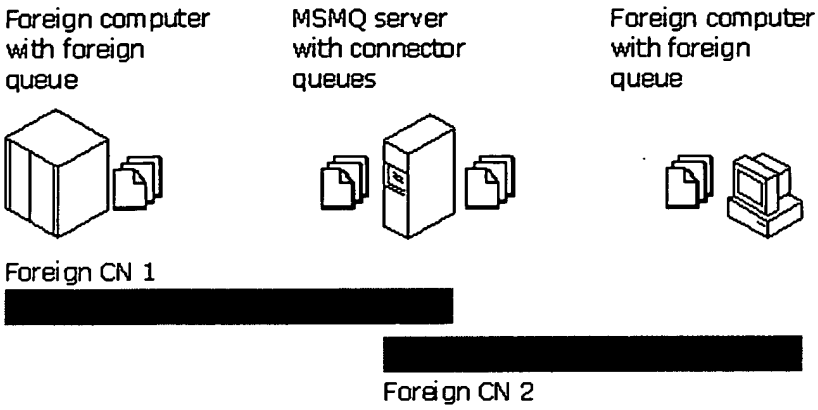

Figure 1.3

Note MSMQ connector servers and the foreign computers that they support must be in the same site.

To develop an MSMQ connector server, you must obtain the MSMQ Connector SDK.

Windows NT RAS Servers and the MSMQ RAS Connectivity Service

Remote MSMQ independent clients can connect to your MSMQ network through a Windows NT RAS server that is also an MSMQ server, or has the MSMQ RAS Connectivity Service installed. To reduce long distance charges and ensure independent clients always connect to the same site, you should configure a Windows NT RAS server with the MSMQ RAS connectivity service (or install the MSMQ server software on it) in each site. While it is possible to configure MSMQ independent clients to connect to your MSMQ network through a RAS server in any site, this configuration does not provide the same benefits.

MSMQ servers (PEC, PSC, BSC, and routing servers) that are also RAS servers must use only one IP CN. If the MSMQ server is configured to use more than one IP CN, automatic site and CN recognition may fail. As a result, MSMQ independent clients that connect to your MSMQ enterprise through RAS may have poor messaging performance, or may not be able to send or receive messages.

Note MSMQ independent clients must not be configured to use NetBEUI protocol over RAS. Configuring an independent client to use NetBEUI over RAS causes the client name to be resolved to the server's IP addresses, rather than the client's address.

RAS support for NetBEUI is disabled by default. To view your RAS protocol settings, run Control Panel, double-click **Network**, click the **Services** tab, click **Remote Access Service**, click **Properties**, click the appropriate modem, and then click **Network**.

↑ Top of page

MSMQ Independent Clients

MSMQ independent client software can be installed on computers running Windows 95, Windows NT Workstation version 4.0 or later, and Windows NT Server version 4.0 or later. MSMQ independent clients can create and modify queues locally, and can send and receive messages, just as MSMQ servers can. MSMQ independent clients can create queues and store messages on the local computer, without synchronous access to an MSMQ server. The primary differences between MSMQ independent clients and MSMQ servers are that independent clients do not have the intermediate store-and-forward capability of MSMQ servers, nor do they store information from the distributed MSMQ database.

In addition to the basic MSMQ files, you can install the MSMQ SDK on MSMQ independent clients.

You can also install the MSMQ Explorer on MSMQ independent clients running under Windows NT Workstation or Windows NT Server so that you can administer your MSMQ enterprise remotely from computers running Windows NT Workstation.

Client Session Limitations

MSMQ clients (both independent and dependent) are limited to a maximum of ten concurrent sessions with other MSMQ clients.

Disconnected Messaging

MSMQ independent clients can send messages to public queues while disconnected from the network. The *disconnect* can be a brief interruption in a network server or the mobile use of a laptop or portable computer.

MSMQ supports disconnected messaging automatically, without additional application design or network configuration. However, you cannot install MSMQ independent clients while the PSC is disconnected because MSMQ Setup must have access to the MQIS on the PSC.

Connecting Through Multiple Sites

MSMQ independent clients can move to other CNs and sites without manual reconfiguration. When a client connects to the new site, all messages that were sent to the client while it was disconnected are rerouted from the client's previous site to the client at the new site. Messages that are sent to the client while it is connected at the new site are routed directly to the client at the new site. However, if the client specifies a new site before disconnecting from the network, messages that are sent to the client while the client is disconnected are routed directly to the new site before the client reconnects.

When an MSMQ independent client moves to a different site, its InRS and OutRS settings are ignored (when routing messages) until the client returns to its original site. You can reconfigure InRS and OutRS settings for the independent client while it is out of its original site, but the InRS and Out RS settings are still ignored. When the independent client is returned to its original site, InRS and OutRS settings are again used when routing messages.

MSMQ does not support mobile servers. To move a server to another CN, simply use MSMQ Explorer to change the server's CN assignments. To move a server to another site you must uninstall the server and then reinstall it in the new site.

How Automatic Site Recognition Works

On MSMQ independent clients, the MSMQ service sends out a broadcast when it starts, and monitors all replies. If a site controller other than the independent client's current site controller replies, and if the new site controller can communicate with the site controller in the independent client's original site, the independent client connects to the new site.

If you have more than one site controller in a broadcast segment (for example, in an MSMQ lab) more than one site controller can reply to the independent client's broadcast. If one of the site controllers that replies is the controller for the independent client's current site, the independent client does not connect to another site.

For information on how to use the MSMQ option in Control Panel to preset your new site, see "Preparing to Travel to a New Site" in Chapter 3, "Managing Your MSMQ Enterprise."

Connecting Through a RAS Server

MSMQ independent clients can connect to your MSMQ network through a Windows NT RAS server that is also an MSMQ server, or a Windows NT RAS server that is configured with the MSMQ RAS connectivity service.

MSMQ independent clients can connect to the MSMQ enterprise through a RAS server (included with Windows NT Server), even if the RAS server belongs to a site or CNs that differ from the independent client's default site or CNs.

MSMQ independent clients connected to your MSMQ network through a RAS server have full MSMQ independent client functionality.

Note You cannot sequentially dial in to two different sites over a RAS line without first restarting the computer, or, under the Windows NT operating system, stopping and then restarting the MSMQ service.

↑ Top of page

MSMQ Dependent Clients

MSMQ dependent clients function much like MSMQ independent clients; however they cannot function without

- synchronous access to a PEC, PSC, BSC, or MSMQ routing server (referred to as the dependent client's *supporting server*). MSMQ dependent clients rely on their assigned server to perform all standard MSMQ functions on their behalf (such as creating queues, sending messages, and receiving messages).

MSMQ dependent clients can be installed on computers running Windows 95, and Intel-compatible computers running Windows NT Workstation or Windows NT Server. (MSMQ dependent clients cannot be installed on Alpha computers running Windows NT). MSMQ servers can support up to 15 dependent clients.

The following diagram shows the logical configuration in which MSMQ dependent clients function:

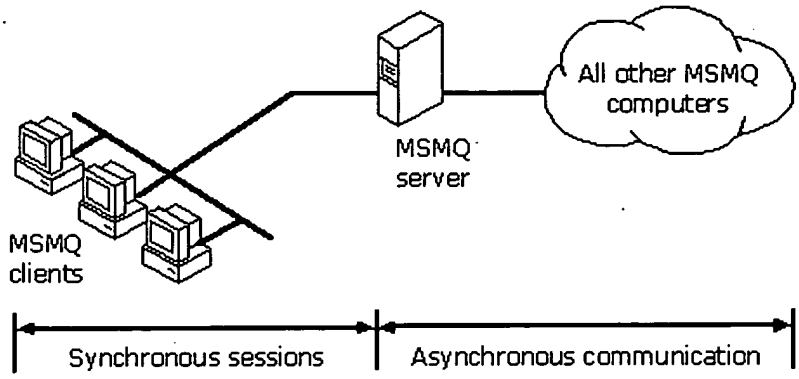

Figure 1.4

This configuration provides the following advantages:

- MSMQ dependent clients on computers running Windows 95 can send and receive transactional messages.
- Server disk space is used to store messages, reducing dependent client hardware requirements. Should more resources be required (memory or hard disk space) fewer computers must be upgraded.
- With fewer points of administration you have fewer computers to back up and fewer journal and dead-letter queues to monitor.

This configuration also has the following limitations:

- Because the MSMQ service runs on the supporting server, encrypted messages sent to or received by MSMQ dependent clients travel between the dependent client and supporting server in an unencrypted format.
- Because MSMQ dependent clients run the MS DTC proxy, they cannot transact MSMQ functions with transactional resources on the dependent client computer; they can transact MSMQ functions only with transactional resources on the supporting server.
- Because MSMQ dependent clients do not run the MSMQ service, they do not appear in MSMQ Explorer. However, when you view the properties of a server, the Dependent Clients tab displays which dependent clients the server supports. You can also use MSMQ Explorer to view the creation of queues and messaging activity for dependent clients on the dependent client's supporting server.

Note An MSMQ dependent client and its assigned server do not have to be in the same site. However, because all of the dependent client's communication goes through the server, if the communication to the server is slow or expensive, all dependent client communication will also be slow or expensive.

For information on the message time-to-reach-queue property, see the MSMQ SDK.

Client Session Limitations

MSMQ clients (both independent and dependent) are limited to a maximum of ten concurrent sessions with other MSMQ clients.

↑ Top of page

Routing

MSMQ establishes a direct connection (a session) using the underlying protocol, if possible. When a direct connection is not possible or not allowed, MSMQ uses its own routing system. MSMQ routing occurs when one or more of the following conditions exist:

- When a session cannot be established between the sender and the receiver (for example, the source and target computers do not share a common CN or the target computer is offline).
- When InRSs or OutRSs are defined for the sender or receiver.
- When messages must travel between two sites, and one or both sites have a site gate defined.

MSMQ servers make two assumptions about your MSMQ network: *Intra-site routing* is fast and inexpensive, while *inter-site routing* is slow and expensive.

Intra-site Routing

Intra-site routing is the process of routing messages within a site. MSMQ measures intra-site routing in *hops*, which is the number of MSMQ servers (PEC, PSCs, BSCs, or MSMQ routing servers) a message must pass through before reaching its destination. MSMQ always chooses the shortest available path when routing a message (unless routing restrictions have been applied by an administrator).

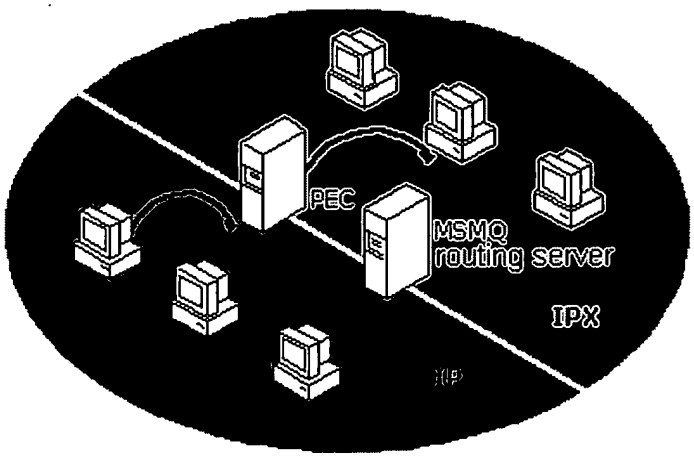

Figure 1.5

In Figure 1.5, any message sent from a computer in the IP CN to a computer in the IPX CN must make two hops: one hop to get to one of the MSMQ servers (assuming they both connect to both CNs) and one hop to get to the target computer. Because each hop is within a site, the cost of each hop is minimal.

The MSMQ intra-site routing system ensures that messages sent between two computers are delivered even if the two computers are never online at the same time.

For information on configuring intra-site and inter-site routing restrictions, see "Session Concentration" later in this chapter.

Inter-site Routing

Inter-site routing is the process of routing messages between sites. MSMQ calculates inter-site routing based on a cost defined by the administrator.

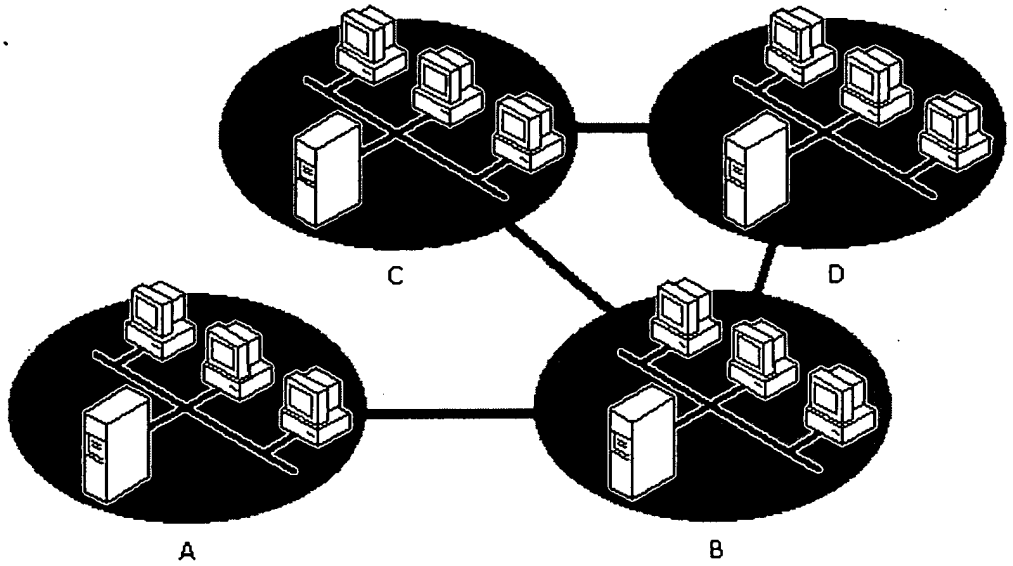

Figure 1.6

The route a message takes to get from site A to site C depends on the cost of the site links. If the cost associated with the A-B site link is 3, and the cost associated with the other three site links (B-C, B-D, and C-D) is 1, messages routed from site A to C will always travel from site A to site B and then to site C. However, if the cost associated with the A-B and B-C site links is 3, and the cost associated to the C-D and B-D site links is 1, messages routed from site A to C will always travel from site A to site B, to site D, and then to site C. If the B-C link, B-D link, or the C-D link is down, messages will be routed using any available link.

For more information on defining the cost of inter-site routing while installing a PSC, see "Installing a PSC" in Chapter 2, "Installing MSMQ." For more information on changing the cost of inter-site routing between existing sites, see MSMQ Explorer Help.

Routing Between Enterprises

MSMQ-based applications can send messages to and receive messages from other enterprises. This allows MSMQ-based applications to communicate over the Internet.

Note MSMQ communicates over TCP port 1801, registered with the Internet Assigned Numbers Authority (IANA). MSMQ-based applications can communicate over properly configured firewalls that allow communication over that port.

For more information on writing MSMQ-based applications that send messages between enterprises, see the MSMQ SDK.

Full Enterprise Connectivity

An MSMQ site is considered to be fully connected if it is possible to route messages, either directly or using MSMQ servers (PEC, PSCs, BSCs, or MSMQ routing servers), between every computer in the site, without having to go to another site. For example, a site does not have full connectivity if it has two CNs, and the only computer that connects the two CNs is an MSMQ independent client.

An MSMQ enterprise is considered to be fully connected if sites are fully connected.

Session Concentration

Session concentration can be used to:

- Reduce sessions within a site
- Reduce sessions between sites

Unnecessary sessions can increase connection and bandwidth costs. MSMQ session concentration reduces these costs by funneling connections through MSMQ servers.

MSMQ supports two types of session concentration: intra-site and inter-site. Intra-site session concentration typically reduces network bandwidth use within a site. Inter-site session concentration typically reduces the number of sessions

between sites.

By loading specific servers with more independent clients, you can use session concentration to manually load or tune your MSMQ network. For example, all MSMQ independent clients in one department can be configured to send and receive all messages through a specific server or set of servers.

Session Concentration Within a Site

Intra-site session concentration is done by configuring MSMQ independent clients to use between one and three dedicated MSMQ servers, (InRSs and OutRSs). If an independent client is configured to use an OutRS, every outgoing MSMQ message sent by the independent client is routed to the OutRS. Likewise, if an independent client is configured to use an InRS, every message sent to the independent client is routed through the InRS.

Only MSMQ independent clients can be configured to have either InRSs or OutRSs; MSMQ dependent clients and MSMQ servers cannot be configured to use InRSs or OutRSs. By default, MSMQ independent clients are not configured to use InRSs or OutRSs.

The PEC, PSCs, BSCs, and MSMQ routing servers can all be used as InRSs and OutRSs. The same MSMQ server can be used as an independent client's InRS and OutRS. However, the InRSs and OutRSs must be in the independent client's original site, and must have at least one CN in common. When a user travels to another site with an independent client, InRS and OutRS settings are disabled until the independent client is returned to its original site.

The cost of session concentration is one additional hop for each message and subsequent additional load on the server. The enterprise administrator must determine the best tradeoff between session concentration versus hops and load balancing based on the configuration and network load.

Intra-site session concentration typically reduces network bandwidth usage. For example, if you use a star topology within a site, and each computer usually communicates directly with every other computer, you can greatly reduce your bandwidth usage by using an MSMQ-based application and configuring each MSMQ independent client to use the PSC as its InRS and OutRS.

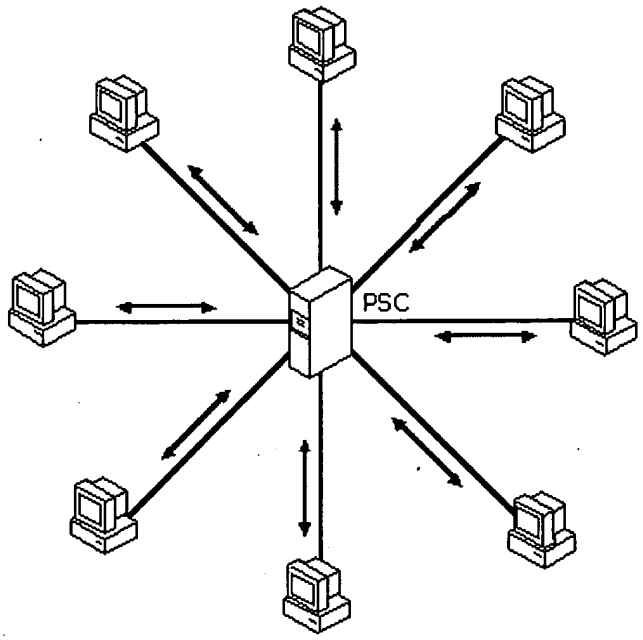

Figure 1.7

In Figure 1.7 the PSC is being used as both an InRS and an OutRS for each MSMQ independent client. This reduces the total number of possible sessions from 36 to 8.

Because computers configured with InRSs and OutRSs are dependent on the MSMQ server, it is preferable to assign more than one InRS or OutRS to independent clients, to provide failure recovery.

For information on how to specify an independent client's InRS or OutRS, see MSMQ Explorer Help.

Session Concentration Between Sites

Inter-site session concentration is done by establishing site gates. If a site is configured to use a site gate, every MSMQ message sent between computers in different sites must be routed through the site gate. In the following figure, the routing topology beyond the site is transparent to the computers within the site, with the exception of the site gate. Thus, the message route is simplified. By default, sites do not use site gates.

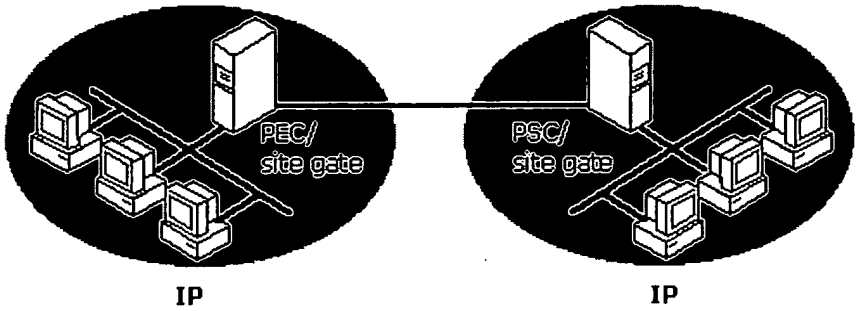

Figure 1.8

In Figure 1.8, site-to-site sessions are reduced. Because each independent client must route messages to its site gate, only the PEC and PSC can establish a session that spans the site.

The following are the requirements for site gates:

- The computer must belong to a site to be a site gate for the site
- The computer must be able to connect to each neighboring site

As long as the computer meets these requirements, the computer can be a PEC, PSC, BSC, or MSMQ routing server.

Because sites configured with site gates are dependent on the site gate for all inter-site transmission of MSMQ messages, you should assign more than one site gate to a site to provide load balancing and failure recovery. There is no limit to the number of servers you can assign as site gates.

For information on how to specify a site's site gate, see MSMQ Explorer Help.

Queue and Message Priority

MSMQ routes messages to public queues based on the sum of each message's message priority and the message's destination queue priority. MSMQ routes messages to private queues base on only the message's message priority. Messages with the highest numerical priority are routed first.

Queue priority (called the *base priority*) for public queues can be set by any MSMQ-based application with write permissions for the queue. The priority can be set at any time. The queue priority can be set from -32768 to 32767. The default priority for public queues is 0. Private queues do not support queue priority.

MSMQ-based applications can send messages with a priority of 0 through 7, 7 being the highest priority.

For more information on changing the base priority of a queue from your MSMQ-based application, see the MSMQ SDK. For more information on using MSMQ Explorer to change the base priority of a queue, see MSMQ Explorer Help.

↑ Top of page

The MSMQ Information Store

The MSMQ information store (MQIS) is a distributed database. MSMQ stores the following information in the MQIS:

- Enterprise topology (such as sites, CNs, and InRS/OutRS assignments)
- Enterprise settings (such as enterprise name, PEC name, and default-replication intervals)
- Computer information
- Queue information

SQL Server Requirement for the MQIS

- MSMQ controller servers (the PEC, PSCs, and BSCs) use a Microsoft SQL Server version 6.5 database to store the MQIS. You can purchase SQL Server 6.5 for each MSMQ controller server, or you can install a limited version of SQL Server 6.5 when you install each controller server.

For information on MSMQ SQL Server requirements, see "Microsoft SQL Server Requirements" in Chapter 2, "Installing MSMQ."

No two computers in your MSMQ network can have the same computer name (also called "friendly" computer names), even if the computers are on separate, non-connected networks.

Everyone has full read access to the MQIS so that all MSMQ users can search for queues within your enterprise. However, access to those queues can still be controlled using a combination of Windows NT security and signed messages.

Note Although the properties of each MSMQ queue reside in the MQIS, the contents of the queues (the messages themselves) reside in memory-mapped files on MSMQ independent clients and servers.

For more information on access control, see "Access Control" in Chapter 5, "Securing Your MSMQ Enterprise." For more information about how MSMQ stores messages, see "Message Delivery" later in this chapter.

MQIS Replication and Ownership

Although the MQIS database is replicated between sites, different servers own and control different sets of data. The MQIS database on the PEC contains the master copy of the enterprise, site, site link, CN, and user settings, and a master copy of its site's computers and queues. The MQIS database on PSCs contains a master copy of their site's computers and queues.

Each PSC MQIS (including the PEC MQIS) also contains a copy of the MQIS information owned by other sites. Each BSC contains a replicated MQIS database from the PSC (or PEC) in its site. The PSCs and the PEC have write access to the MQIS information they own, and read-only access to all replicated information that they receive from the PEC or other PSCs. Each BSC has read-only access to its replicated MQIS databases.

Replication Mechanics

Data is replicated from the owner directly to the other sites using MSMQ messages and private queues. A PEC replicates the enterprise and its site-specific data directly to other PSCs. Each PSC replicates its site-specific settings directly to the PEC. This is called *inter-site replication*. Each PSC (and the PEC) replicates the information it owns and the information it receives from other PSCs to the BSC at its site. This is called *intra-site replication*. PSCs (including the PEC) never replicate information to BSCs in other sites.

For example, in Figure 1.9, the MQIS database on the PEC at site A contains the master database for enterprise, site, site link, and CN settings; the master database for site A computer and queue settings; and replicated data owned by sites B and C. The MQIS database on the PSC at Site B contains the master database for site B computer and queue settings; replicated computer and queue settings from sites C and A; and replicated site, site link, CN, and enterprise settings from the PEC. The MQIS database on the PSC at Site C contains the master database for site C computer and queue settings; replicated computer and queue settings from sites B and A; and replicated site, site link, CN, and enterprise settings from the PEC.

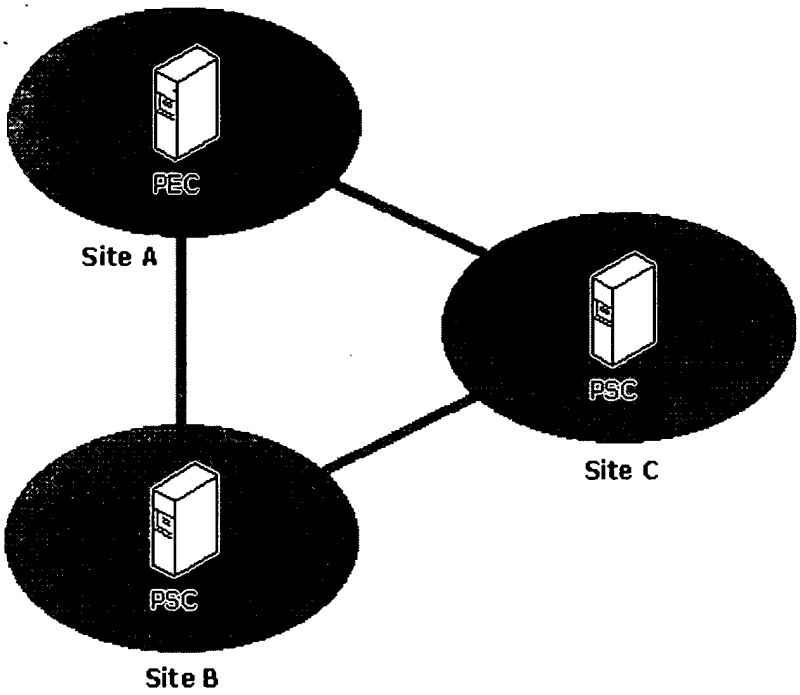

Figure 1.9

Because intra-site replication is quicker and cheaper than inter-site replication, it takes place more often. By default, intra-site replication occurs every 2 seconds, while inter-site replication occurs every 10 seconds. However, the database is immediately replicated whenever the site topology changes, such as when a site is added.

For information on how to change MSMQ replication intervals, see MSMQ Explorer Help.

Top of page

Queues

All MSMQ queues, regardless of their function, can be manipulated with the same MSMQ functions. This includes the private and public queues used to send and receive messages, and the special journal, dead letter, transactional dead letter, administration, and report queues used to add functionality to your MSMQ applications. Each of the queues described in the following sections is simply a standard MSMQ queue used for a specific purpose.

MSMQ Explorer uses a variety of icons to display the different types of queues. To see a table that defines the computer, queue, and message icons, click **Help Topics** on the MSMQ Explorer **Help** menu, double-click **Overview**, and then double-click **MSMQ Explorer Icons**.

Public Queues and Private Queues

Public queues are those published in the MQIS. All public queues are replicated throughout the enterprise, and can therefore be located by any computer within the enterprise.

Private queues are not published in the MQIS, and therefore do not add to the MQIS replication load. Private queues can be accessed only by applications that have access to the full path or format name of the queue. In the default view, MSMQ Explorer does not display private queues.

For more information on viewing private queues, see MSMQ Explorer Help.

Journal Queues

The process of logging or storing a copy of a message in a queue is called *journaling*. Journal messages are stored in a queue called *Journal*.

Two types of journaling are available: source and target. *Source journaling* is the process of storing a copy of an outgoing message. It is configured on a message basis, and is therefore a property set by the sending application.

When source journaling is enabled for a non-transactional message, a copy of the message is placed in the *source-journal queue* of the sending computer either when the message arrives at the target queue or when the message reaches an MSMQ server that is one hop closer to the target computer. When source journaling is enabled for a transactional message, a copy of the message is placed in the *source-journal queue* of the sending computer when the message arrives at the target queue. A copy of the message is never stored in a source-journal queue on a server that forwarded the message.

Source journaling messages for MSMQ dependent clients are placed in the source-journal queue on the supporting MSMQ server. In MSMQ Explorer, the source-journal queue is displayed underneath the computer.

Target journaling is the process of storing a copy of incoming messages. It is configured on a queue basis. When target journaling is enabled, a copy of each incoming message is placed in the *target-journal queue* when the message is removed (read) from the target queue. A target-journal queue (Journal) is created for each queue when the queue is created. MSMQ Explorer displays target-journal queues under each public queue.

For information on enabling target journaling, see MSMQ Explorer Help. For information on enabling source journaling, see the MSMQ SDK.

Dead-Letter Queues

Messages that have expired or are undeliverable are stored in dead-letter queues. A non-transactional message that cannot reach the destination application is stored in a dead-letter queue on the computer on which the message expired (or failed): this can be the source computer, the destination computer, or an MSMQ server that is forwarding the message.

Dead-letter queues are created for each independent client and server on your MSMQ network when MSMQ is installed on that computer. These queues are displayed under the computer in MSMQ Explorer.

Messages are undeliverable when:

- The destination queue is unknown
- The message has exceeded its maximum number of hops
- The message time-to-be-delivered (TTBD) has expired
- The message time-to-reach-queue (TTRQ) has expired
- The queue's quota has been reached
- A non-transactional message is sent to a transactional queue and the queue properties cannot be accessed when the message is sent

Messages are not stored in dead-letter queues when the message sender is not allowed to send messages to the destination queue. Messages in dead-letter and journal queues are always recoverable.

For more information on message properties (such as TTBD and TTRQ), see the MSMQ SDK. For more information on recoverable messages, see "Message Delivery" later in this chapter. For more information on queue and computer quotas, see "Queue and Computer Quotas" later in this chapter.

Transaction Dead-Letter Queues

A transactional message that cannot reach the destination application is stored in the transaction dead-letter queue on the source computer. Transaction dead-letter queues are created for each independent client and server on your MSMQ network when MSMQ is installed on that computer. MSMQ Explorer displays these queues as **Xact Dead Letter**, under the computer.

Transactional messages cannot reach the destination application when:

- The sending transaction is ended (for example, because another transaction participant failed).
- The message failed to reach the destination queue during the specified Time-To-Reach-Queue interval.
- The message is rejected by the destination queue (due to security or quota settings, or because the destination queue

is not a transactional queue).

- The message reached the queue, but was not successfully received from the destination queue during the specified Time-To-Reach-Queue interval.
- The receiving transaction failed.

For more information on transactional messages, see "Transactional Messaging" later in this chapter.

Administration Queues

Administration queues contain acknowledgment messages generated by MSMQ for messages you send. These messages indicate that the messages you sent either arrived (a positive acknowledgment) or that an error occurred before the message could be retrieved (a negative acknowledgment). Positive acknowledgments indicate whether or not the message was retrieved by the target application. Negative acknowledgments indicate why the message was not retrieved.

The application programmer decides whether to use acknowledgments. If they are used, the application programmer must specify an administration queue in the source application and the type of acknowledgment messages to send to the queue.

Note Negative acknowledgments (NACKs) for encrypted messages are sent without the message body. Because only the destination computer can decrypt the message body, the message body is not useful on other computers.

For more information on administration queues, see the MSMQ SDK.

Report Queues

Report queues contain report messages that indicate the message's route through your enterprise. Report queues can be used when:

- Sending test messages
- Tracking message routes for a specific application

For more information on sending test messages and enabling message route tracking, see "Tracking and Monitoring Messages" in Chapter 4, "Monitoring Your MSMQ Enterprise."

System Queues

MSMQ uses up to six system queues. All six system queues are implemented as private queues. Each queue and its use is described in the following table.

Table 1.1

System Queue	Use
admin_queue$	Storing administrative messages (such as test messages)
explorer_response_queue	Storing administrative response messages (except MQPing response messages)
mqping response	Storing returned MQPing requests
mqis_queue$	Storing MQIS replication messages
notify_queue$	Storing notification messages. These messages inform a computer about a change in its properties or a change to one of its queues (including creation and deletion of queues).
order_queue$	Tracking transactional messages that require in-order-delivery

The MSMQ system queues are not affected by a computer's message quota.

System queues cannot be deleted. Those users with Administrative privileges on their local computer have full control of the system queues on their computer except the Delete permission. Everyone has Get Properties, Get Security, and

- Send permission for system queues.

Top of page

Message Delivery

MSMQ messages can contain text or binary information in the message body. The contents of the message body are specified by the sending application—MSMQ handles the message delivery but does not alter the message contents. MSMQ messages can be no larger than 4 megabytes (MB).

To choose the appropriate delivery method for an MSMQ-based application, and to calculate required disk space on MSMQ independent clients and servers, you must understand:

- Message storage details
- The two delivery types available
- Journaling queues
- Dead-letter queues

To install and administer an MSMQ network, you must understand:

- The two message delivery types available
- Message storage details
- Messaging limitations
- Queue and computer quotas
- Transactional messaging

MSMQ Explorer uses a variety of icons to display the different types of messages. To see a table that defines the computer, queue, and message icons, click **Help Topics** on the MSMQ Explorer **Help** menu, double-click **Overview**, and then double-click **MSMQ Explorer Icons**.

Express and Recoverable Delivery

MSMQ supports two delivery methods: express and recoverable. Choosing between express and recoverable delivery is a matter of trading performance and resource use for reliability and failure recovery.

In general, express messages use fewer resources and have faster throughput than recoverable messages. However, express messages cannot be recovered if the computer storing the memory-mapped message files fails. Recoverable messages use more resources and have slower throughput than express messages, but can be recovered no matter which computer fails.

For example, if an MSMQ-based application is sending express messages to a queue through an MSMQ routing server, and the network link between the MSMQ routing server and the destination computer fails, the messages will continue to be stored in memory on the MSMQ routing server. If the MSMQ routing server is shut down before the network link is restored, the express messages are lost. However, if recoverable messages are used, the messages are not lost and are delivered after the MSMQ routing server and network link are brought back online.

The sending application, not MSMQ, specifies the delivery method.

Message Storage Details

While queue information is stored in the MQIS, messages are stored in memory-mapped files (.mq files) on the sending, receiving, or intermediate MSMQ routing server's hard disk. MSMQ allocates disk space for both express and recoverable messages in 4 MB files, as needed. For each recoverable message, the write operation is flushed to disk, ensuring the data is recoverable. For express messages, memory-mapped files are written to disk only when the computer has insufficient resources to store the file in memory.

The memory-mapped files are created on the computer sending the message. As messages are forwarded by intermediary store-and-forward servers, they are also stored on those servers in memory-mapped files. If source or

- target journaling is being used, the memory-mapped files are kept on the sending or receiving computers, respectively. MSMQ servers store all messages for the MSMQ dependent clients they support.

The memory-mapped files can be stored on any file system supported by the underlying operating systems. However, because the Windows NT File System (NTFS) provides more reliable file recovery after a power failure or system crash, it is the recommended file system for MSMQ servers.

Messaging Limitations

Messaging throughput is limited by the hardware being used. Processing power, available memory, and network capacity have the greatest affect on messaging throughput. The number of messages each MSMQ independent client and server can store at one time is limited first by the available disk space on the local computer, and second, by the code and data space of the operating system on which MSMQ is running.

When running Windows 95, MSMQ independent clients can store close to 1 gigabyte (GB) of messages. Specifically, MSMQ independent clients can store 1 GB of messages minus the memory consumed by the overhead of storing the message and the memory of other applications using the shared system memory.

When running version 4.0 of Windows NT Workstation or Windows NT Server, MSMQ independent clients and servers can store just under 2 GB of messages (specifically, 2 GB minus 20 MB).

On all platforms, each MSMQ message can have no more than 4 MB of data.

For MSMQ servers supporting multiple MSMQ dependent clients, this limit applies to the server, not each individual client. For example, an MSMQ server supporting three dependent clients can hold no more than a cumulative total of 1.98 GB of messages for the three dependent clients.

Queue and Computer Quotas

Queue and computer quotas specify the cumulative limit for messages in a public queue or computer. The queue and computer quotas are based on size, and can be set independently. When a queue quota is reached, messages can no longer be sent to the queue or computer until one or more messages are removed from the queue. Different error messages are returned, depending on the queue location. Queue quotas can be set on both public queues and journal queues.

MSMQ enforces the computer limit regardless of the number of queues opened or the cumulative queue quotas. For example, if you specify a 10 MB limit for each of the six public queues on a computer, and a 50 MB total limit for a computer, MSMQ will enforce the 50 MB computer limit before each queue quota is reached. However, each queue quota still prevents any one queue from storing more than 10 MB of messages.

Queue and computer quotas can be set only for MSMQ independent clients and servers. Queue and computer quotas set on MSMQ servers that are supporting MSMQ dependent clients apply to the server and all its dependent clients as a whole. For example, if the computer quota is 50 MB, and one dependent client creates 50 MB of messages that are not deliverable, the server and the other dependent clients supported by the server cannot send messages until the total message store drops below the quota.

Queue and computer quotas can be enabled or changed at any time, either programmatically or by using MSMQ Explorer. You must have write permission for a queue or computer to change its quota.

Note Computer quotas do not apply to system queues.

For more information on changing queue and computer quotas programmatically, and for information on error messages returned when a queue or computer quota has been reached, see the MSMQ SDK. For more information on changing queue and computer quotas using MSMQ Explorer, see MSMQ Explorer Help.

Transactional Messaging

MSMQ can be used as a resource manager under the control of Microsoft Distributed Transaction Coordinator (MS DTC). Using MS DTC and MSMQ you can implement transaction-based applications, ensuring that message operations either succeed or fail in conjunction with other OLE-transaction-compliant applications and other MSMQ operations. For example, an MSMQ-based application can send a message and update a database in the same transaction. MS DTC ensures that both actions succeed, or neither are executed. These standard two-phase transactions are called *coordinated transactions* in the MSMQ SDK.

MSMQ also provides a small transaction coordinator that supports only one resource monitor: MSMQ. These single-phase transactions are simply called *transactions* in the MSMQ SDK. Because they do not have the overhead of two-phase commit transactions, these transactions are much faster than transactions coordinated by MS DTC.

Note Computers running Windows 95 and the MSMQ independent client software cannot send or receive transactional messages. However, computers running Windows 95 and the MSMQ dependent client software can send and receive transactional messages using the MS DTC proxy on the MSMQ dependent client. The MS DTC proxy is always installed on MSMQ dependent clients.

Using the MSMQ transaction parameter you can:

- Transact the sending or receiving of any message with an action in any other OLE-compliant resource (for example, to update a SQL database and send a message) and transact the sending or receipt of multiple messages
- Ensure that a message is delivered only once (also called exactly once delivery)
- Ensure all messages sent from one computer to another are delivered in order (also called in order delivery)
- Use receive acknowledgments (ACKs) to confirm that the received transaction was successfully committed or to confirm that messages reached or were retrieved from the destination queue

Transactional messages are standard MSMQ messages with an additional flag set. However, transactional messages can be sent only to transactional queues. Non-transactional messages cannot be sent to transactional queues, and transactional messages cannot be sent to non-transactional queues.

For more information on using transactions with your MSMQ-based applications, see the MSMQ SDK.

↑ Top of page

Programming Options

You can integrate your MSMQ-based application with the MSMQ Exchange connector, the MSMQ MAPI transport provider, and the MSMQ RPC transport. Using the ActiveX™ controls provided by MSMQ and Microsoft Internet Information Server (IIS), you can integrate your MSMQ-based application with Web pages and forms.

You can develop MSMQ-based applications on MSMQ servers and MSMQ independent clients. You cannot develop MSMQ-based applications on MSMQ dependent clients.

MSMQ Exchange Connector

Exchange users can send messages to and receive messages from MSMQ queues using the MSMQ Exchange connector. The Exchange software does not have to be running on other MSMQ dependent clients, independent clients, or servers that send messages to the Exchange clients, nor does the MSMQ software need to be running on the Exchange client computers.

For information on installing and using the MSMQ Exchange connector, see Appendix B, "Installing and Configuring the MSMQ Exchange Connector."

MSMQ MAPI Transport Provider

Any messaging application programming interface (MAPI) client can send and receive MAPI messages using the MSMQ MAPI transport provider.

The MSMQ MAPI transport provider differs from the MSMQ Exchange connector in the following ways:

- The MSMQ MAPI transport must be installed on each MAPI client to allow the client to send and receive messages over MSMQ. In contrast, the MSMQ Exchange connector needs to be installed on only one MSMQ server within a site.
- The MSMQ MAPI transport provider enables any MAPI client to send and receive MSMQ messages, whereas the MSMQ Exchange connector enables only Microsoft Exchange clients to send and receive MSMQ messages.
- Every MAPI client has its own MSMQ queue for incoming messages created by the MSMQ MAPI transport. In contrast, Microsoft Exchange clients receive messages from MSMQ through one queue used by the MSMQ Exchange connector.
- It is more difficult for a group of MAPI clients using the MSMQ MAPI transport to share addresses of MSMQ queues.

- Each MAPI client must store the queue addresses in its own Personal Address Book (PAB). Conversely, Microsoft Exchange clients can obtain MSMQ queue addresses from the shared Exchange Address Book.

MSMQ RPC Transport

With the MSMQ RPC transport you can use MSMQ as a reliable transport for your RPC-based applications.

To use MSMQ as an RPC transport, you must install Windows NT Service Pack 3 on each computer running Windows NT Workstation or Windows NT Server that will send or receive RPC packets over MSMQ. You must install DCOM 95 on each computer running Windows 95 that will send or receive RPC packets over MSMQ.

MSMQ uses authenticated RPC. Authenticated RPC connections must be validated by a Windows NT Server domain controller. If computers access a domain controller over a slow link, RPC connections will start slowly. If the domain controller is offline, computers cannot read messages (sent using the RPC transport) from queues located on other computers.

Internet Connectivity Using ActiveX Contols

Using ActiveX controls provided by MSMQ in conjunction with active server pages, you can integrate your MSMQ-based applications with Web pages and forms.

For information on using ActiveX controls provided by MSMQ in conjunction with Microsoft Active Server, see the MSMQ SDK.

Top of page

Business Scenarios

The following business scenarios show different business situations that can benefit from MSMQ-based applications, and the different programming options available when you use MSMQ.

Note The term *client* in this section refers to a computer that is running MSMQ. The differences between MSMQ dependent clients, MSMQ independent clients, and MSMQ servers affect implementation details, not business scenarios.

Floor Automation

You can use MSMQ to control the automated manufacturing of any product that is produced with a numerically controlled industrial machine that is controlled through a personal computer. In the following example, MSMQ is used to collect orders and manage the production flow by controlling the personal computers that control the industrial machines.

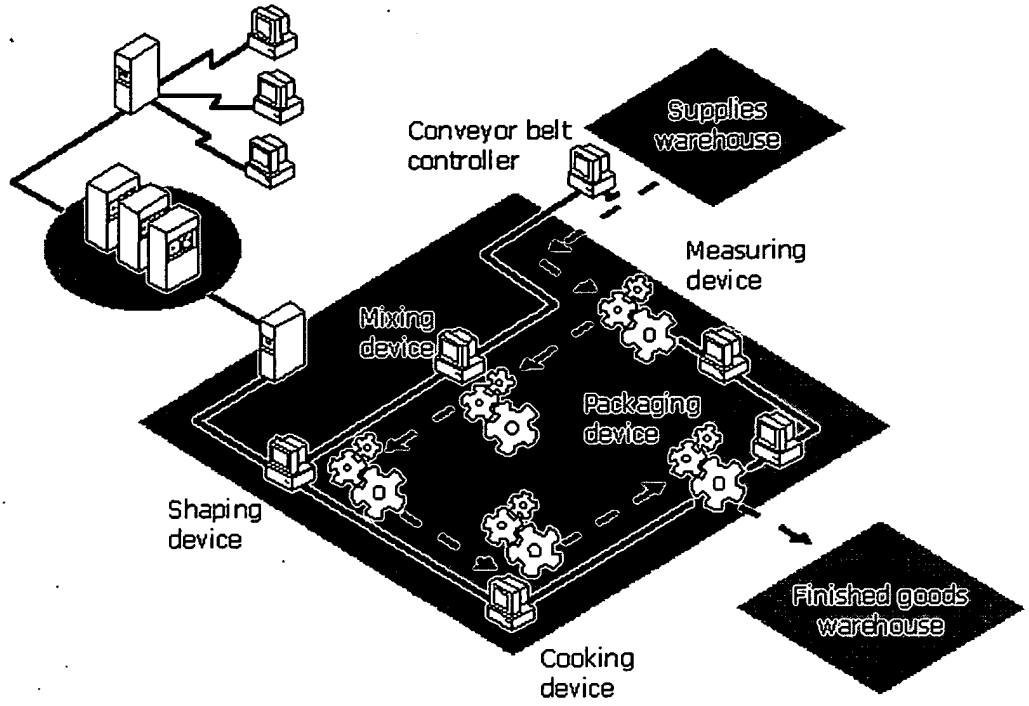

Figure 1.10

The following MSMQ features are critical to the success of floor automation:

- Guaranteed delivery: Ensures that each message is delivered and helps ensure a reliable messaging system.
- Message prioritization: Efficient routing of messages. For example, quality assurance (QA) messages need to be delivered eventually, but shouldn't prevent the efficient delivery of time-critical messages.
- Connectionless messaging: Interactions of various clients, the personal computers controlling the industrial machines, and a mainframe computer as a single system.
- Multiple platform support: Integration of personal computers and legacy systems. For example, an accounting package may reside on an MVS system.
- Performance: Control of real-time production processes, such as controlling cooking temperature.
- Asynchronous delivery: Faster devices can continue working without waiting for slower devices to respond.

Ticket Sales

The following ticket sales scenario uses these MSMQ features:

- Internet connection: The MSMQ-based application can use the ActiveX controls provided by MSMQ, in conjunction with Microsoft Active Server and IIS, to accept orders over the Internet through HTML forms, providing security and multiple-platform support.
- Connectionless messaging: Account updating and ticket shipping can be performed asynchronously.
- Multiple platform support: Integration of personal computers and legacy systems. For example, the Venue/Show server may be a UNIX computer.

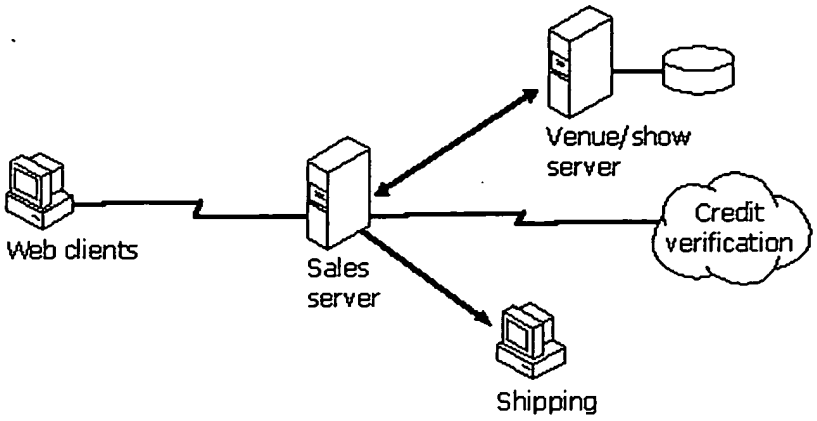

Figure 1.11

In Figure 1.11, multiple Web clients automatically access the ticket sales server. Web clients view HTML pages that provide ticket availability, pricing, and seating diagrams.

Publish and Subscribe

MSMQ can be used to simplify the development of traditional publish and subscribe applications. These applications are unique because although clients do subscribe to receive a particular type of data, they do not have to continuously request the data. Similarly, although a server application must continuously send updated data of a specified type to the client, the server does not have to continuously try to resend the data if the client if offline or unavailable. Traditional publish and subscribe applications include newswire services and financial or stock-market-data subscription services.

The following figure shows a publish and subscribe scenario that keeps sales staff up-to-date on the status of available goods. In this scenario, an MSMQ application could also be used to take orders, and the two systems could be integrated. However, to simplify the details of the publish and subscribe scenario, this scenario covers only the dissemination of information.

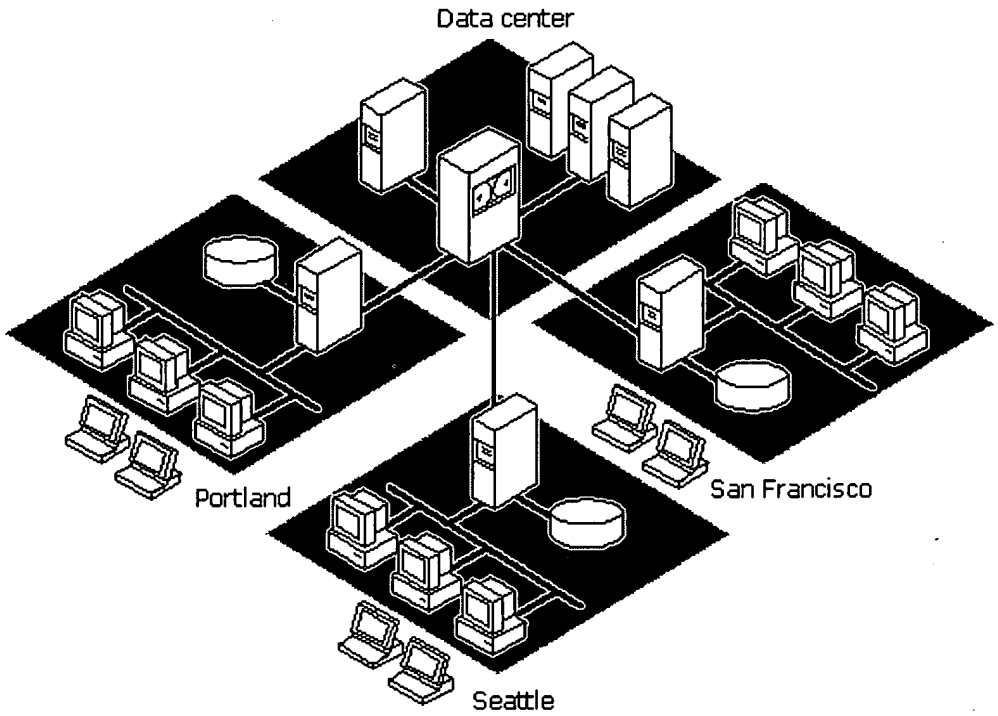

Figure 1.12

In Figure 1.12, the mobile independent clients retrieve data as they work in the field. The regional servers retrieve from the main server only those data types to which the MSMQ independent clients subscribe. Each MSMQ independent client then receives only the data it has subscribed to receive. This reduces both inter-site communication and mobile user's

cellular phone use.

The following MSMQ features are critical to the success of traditional publish and subscribe applications:

- Connectionless messaging and message prioritization: Places the burden of message delivery on MSMQ. The application programmer then does not have to deal with information that expires and is no longer useful, or with information that cannot be delivered in a timely fashion.
- Security: Ensures that the correct information is delivered to the correct customer and ensures that services are not being used without payment.

Banking/ATM

Financial transactions generally require security and transaction-based programming. Additionally, automated teller machine (ATM) transactions require connectionless messaging. In Figure 1.13, each ATM has its own local database. Even if the local, regional, or headquarters database is temporarily offline, the ATM can still process transactions.

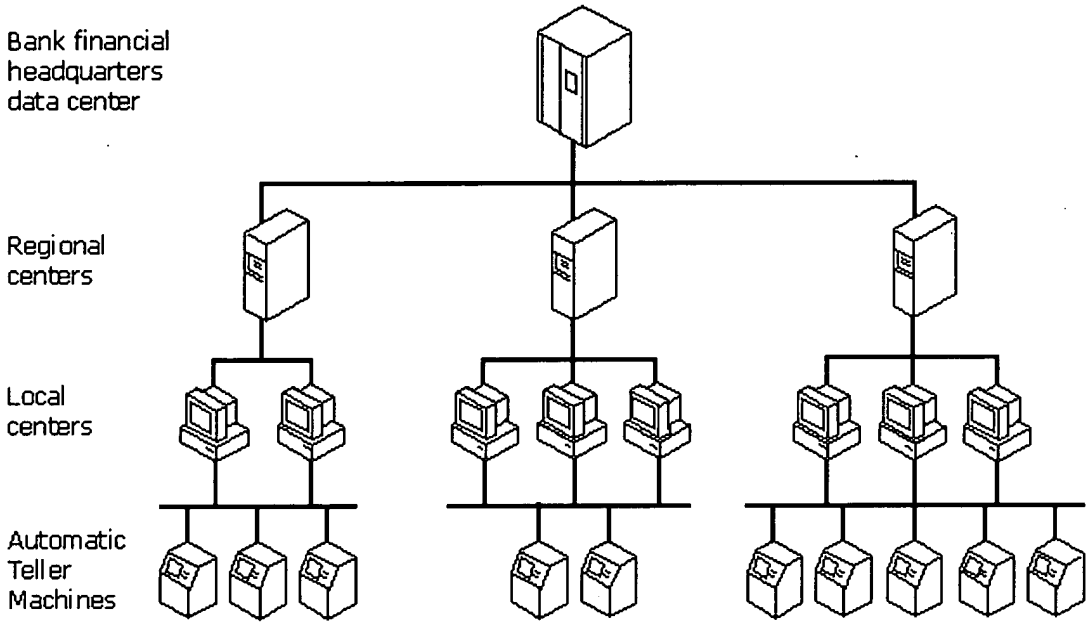

Figure 1.13

The following banking scenario uses these MSMQ features:

- Connectionless messaging: Communication between ATMs and regional centers.
- Transactional messaging: The sending and receiving of messages that occurs synchronously with the updating of the financial databases; messages are delivered only once.
- Security: Ensures the integrity of the system. Would-be thieves cannot impersonate an ATM, and transactions cannot be monitored to steal account numbers and personal identification numbers (PINs).

For more information about implementing an MSMQ-based application that uses transactional messaging, see the MSMQ SDK.

Stock Brokerage

The following stock transaction scenario uses these MSMQ features:

- Security: Message authentication, encryption, and auditing ensure a secure environment.
- Multiple platform support: Because they often must work across disparate systems, messaging allows the different platforms to communicate.
- Transactions (Exactly once delivery): Guaranteed message delivery ensures accurate accounting.

- Connectionless messaging: Account updating and risk assessment can be performed asynchronously.

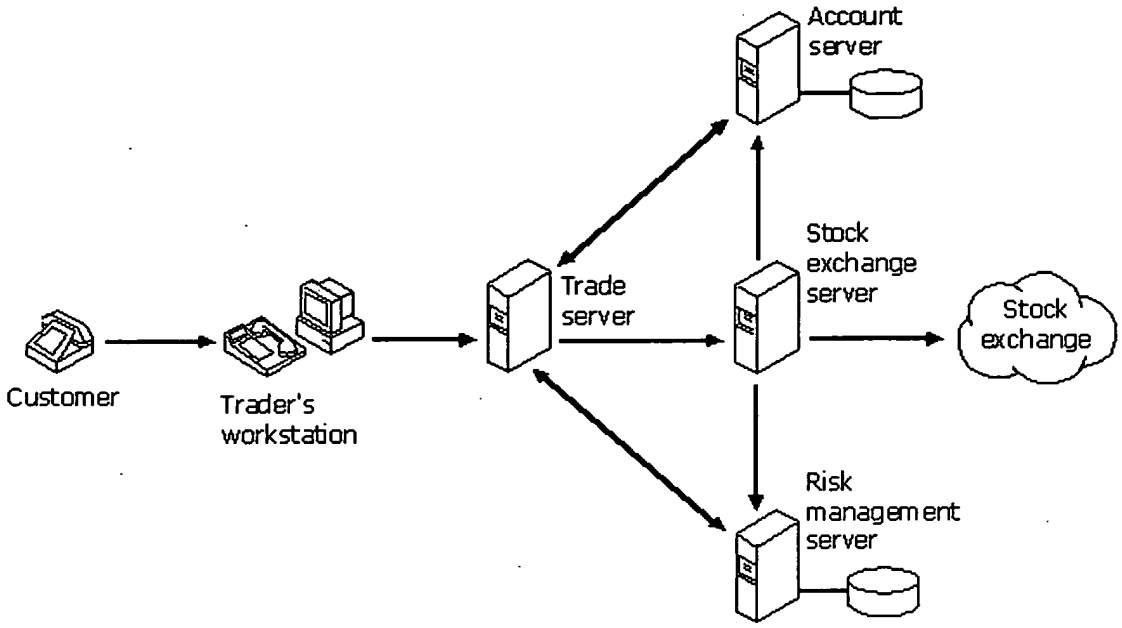

Figure 1.14

In the example shown in Figure 1.14, customers communicate by telephone with traders who make trade requests on behalf of the customers. The trade server performs the following tasks in the following order:

- Verifies the transaction can be completed by querying the account server
- Verifies the transaction with the risk-management server
- Places the trade request with the stock exchange server

The stock exchange server makes the stock transaction and then asynchronously updates the customer account and risk-management server.

You can use the MSMQ RPC transport to access the databases, Visual Basic and the ActiveX controls provided by MSMQ to write client applications, and C to write the core brokerage application.

Intranet Applications

With your MSMQ infrastructure in place, you can implement MSMQ-based applications to integrate internal processes. For example, you can create an MSMQ-based application to allow users to request helpdesk support from various e-mail clients, Exchange forms, and browsers, and then automatically publish resolved issues in Exchange public folders.

This MSMQ-based application might use:

- The MSMQ Exchange connector
- The MSMQ MAPI transport provider
- The ActiveX controls provided by MSMQ, in conjunction with Microsoft Active Server and Internet Information Server (IIS)

Top of page